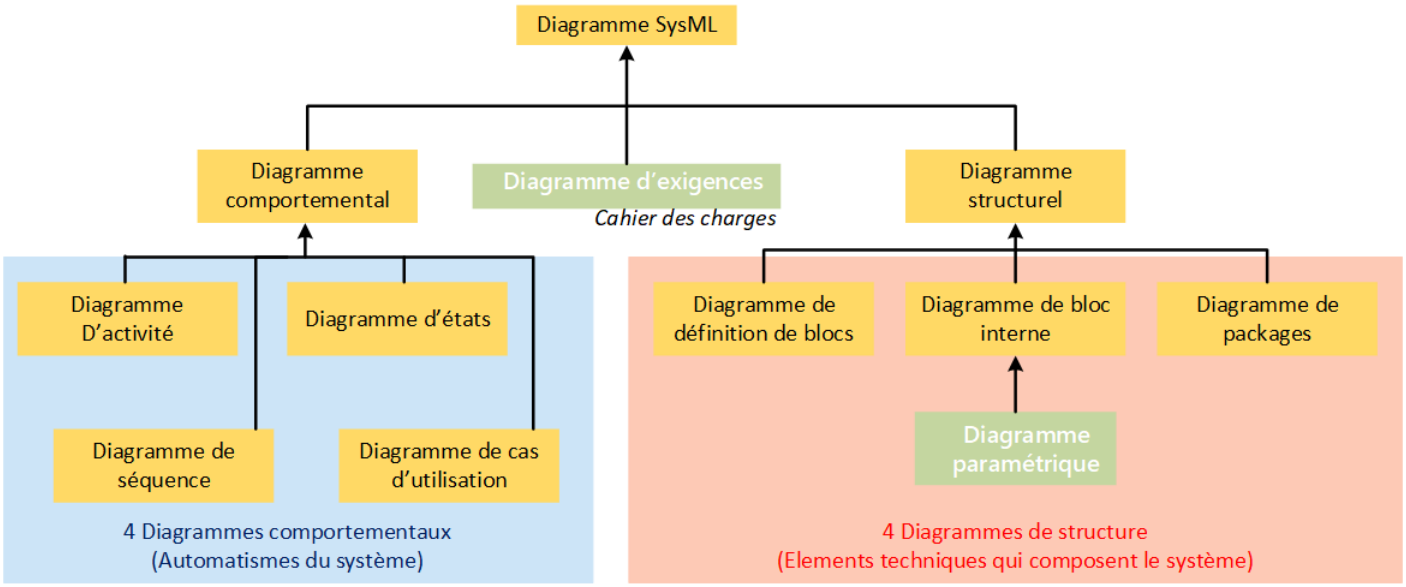
Méthodologie - Diagrammes UML/SYSML

I. Introduction

Voici un résumé des diagrammes présents dans la méthode UML/SysML. Ensemble il serviront à constituer le cahier des charges et peuvent même en faire directement partie.



Si l'on prends le cas, par exemple, d'un radio réveil.

Le besoin exprimé est le suivant :

"*Je veux arriver à l'heure en cours, j'ai donc besoin de me réveiller à l'heure et de partir à l'heure.*"

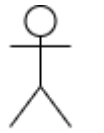
II. Diagramme de cas d'utilisations

Partant de ce besoin là, il est possible de créer le diagramme des cas d'utilisation qui permet de se représenter les besoins attendus par le système à mettre en place.

Pour réaliser le diagramme, il faut se placer du point de vue de l'utilisateur.

La nomenclature est la suivante :

un ***acteur*** est identifié par un stickman.



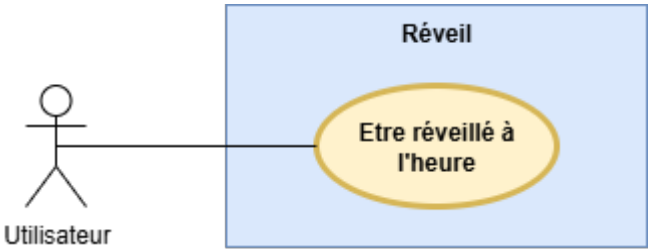
un ***cas d'utilisation*** (fonctionnalité) est illustrée par un ovale.

Une association signifiant "***participe à***" est représentée par une ligne.

Dans le cas du réveil, le fait d'être réveillé à l'heure implique 2 choses :

- Connaître l'heure.
- Avoir un moyen de se réveiller (par un son ou une musique par exemple).

Cela donne le diagramme suivant :



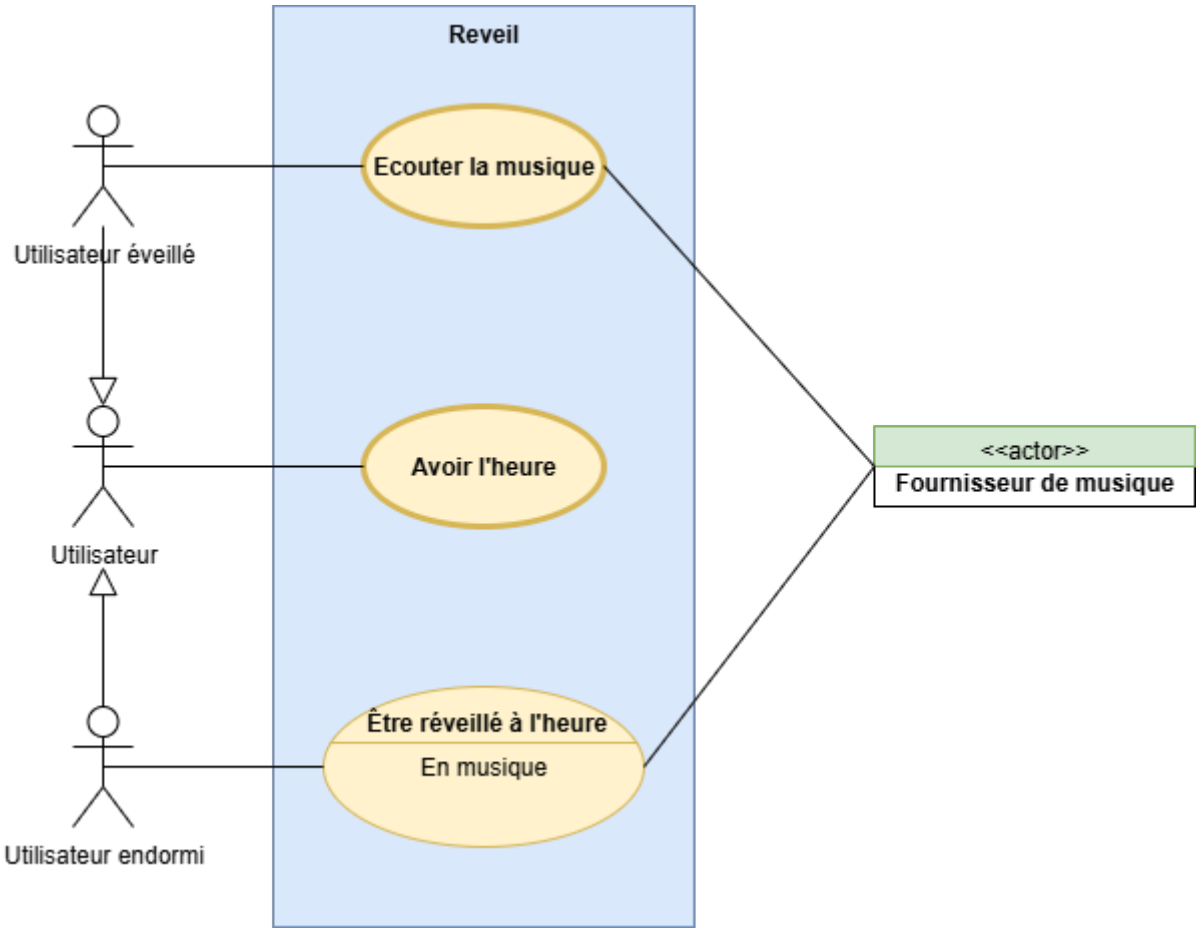
Il est possible de compléter le diagramme avec tous les cas d'usage selon les états. En effet, l'utilisateur pourra être endormi ou réveillé et n'utilisera donc pas les même fonctionnalités. De

plus, la musique pourrait provenir d'un fournisseur de service.

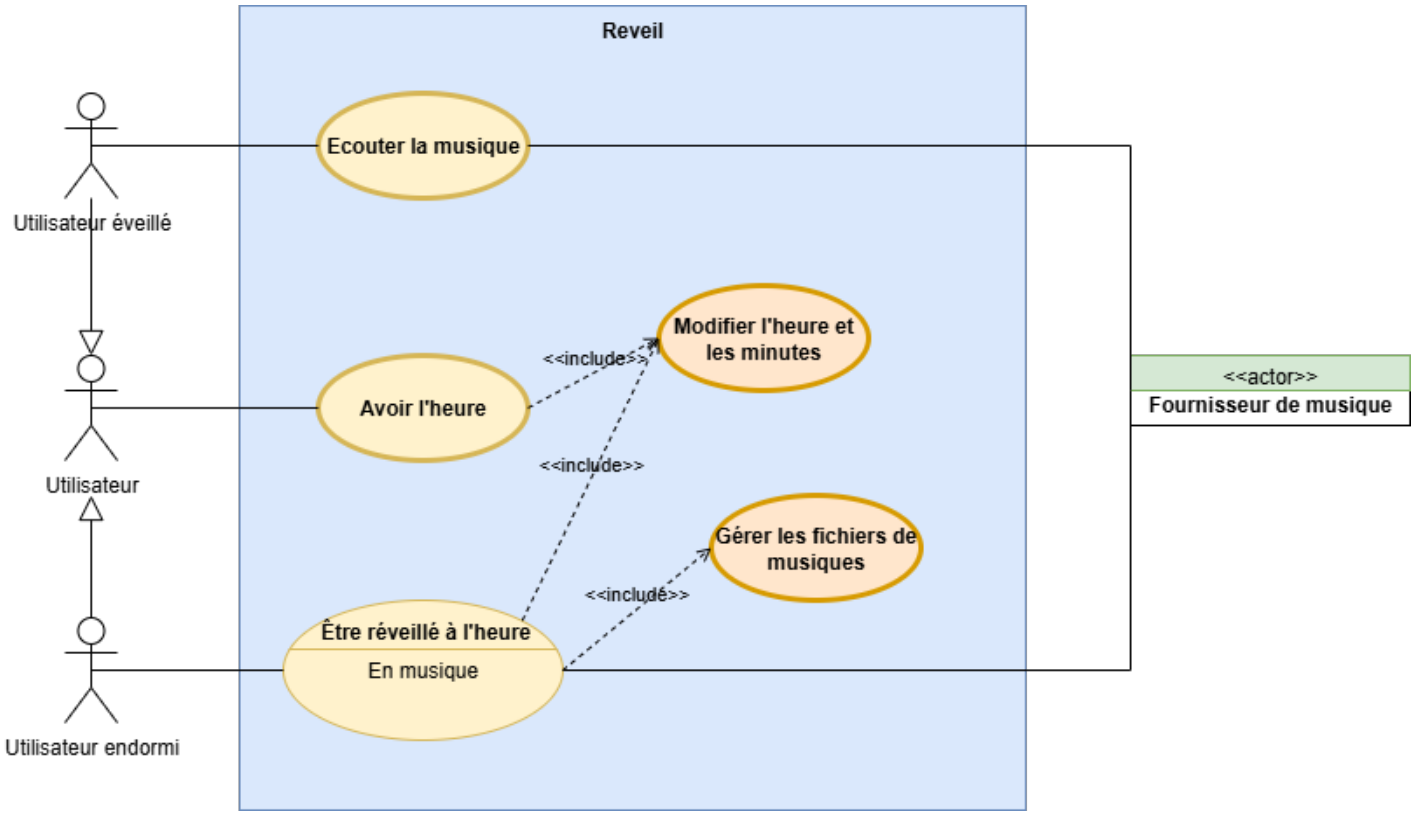
Pour lier les états à un acteur, utiliser une ***généralisation*** représentée par une ligne fléchée depuis le cas enfant vers le cas parent.

Pour indiquer la présence d'***acteurs externes***, il faut ajouter un carré avec le nom de l'acteur. Par convention, les acteurs internes sont à Gauche du schéma et les acteurs externes sont à droite.

Ce qui donne le schéma suivant :



Enfin, l'identification des cas d'utilisation peut mettre en exergue des dépendances ou fonctionnalités implicites. Il faudra alors inclure un cas d'utilisation et le lier à son cas parent avec une ***relation d'inclusion*** matérialisée par une flèche pointillée marquée <<include>>.



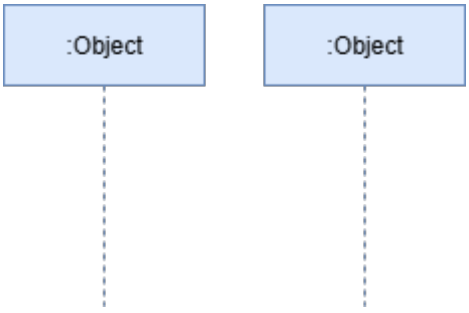
Enfin, d'un besoin peut en découler un autre, auquel cas il faudra créer une ***relation d'extension*** matérialisée par la même flèche pointillée dans l'autre sens avec la mention <<extend>>.

III. Diagramme de séquence

Il faudra également modéliser les séquences d'actions permettant de gérer une fonction. Toujours dans le cas du réveil, quelle est la séquence attendue lors d'un réveil.

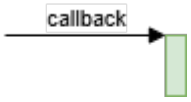
Cela se modélisera de façon séquentielle, dans l'ordre chronologique, du haut vers le bas.

La nomenclature est la suivante :

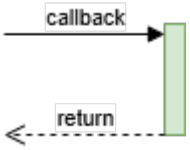


En haut du processus d'échange on indique les ***acteurs***

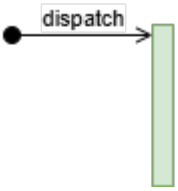
Un ***message synchrone*** est un message en attente de réponse.
Il est modélisé par une flèche trait plein avec tête pleine.



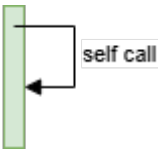
Le ***message retour*** est modélisé par une flèche pointillée.
Ce message est le résultat direct du message précédent.



un ***message asynchrone*** est un message qui n'attends pas de réponse. Il est modélisé par une flèche trait plein avec une tête évidée.



Une boucle, appelée message réflexif est représenté par une flèche pleine bouclée.

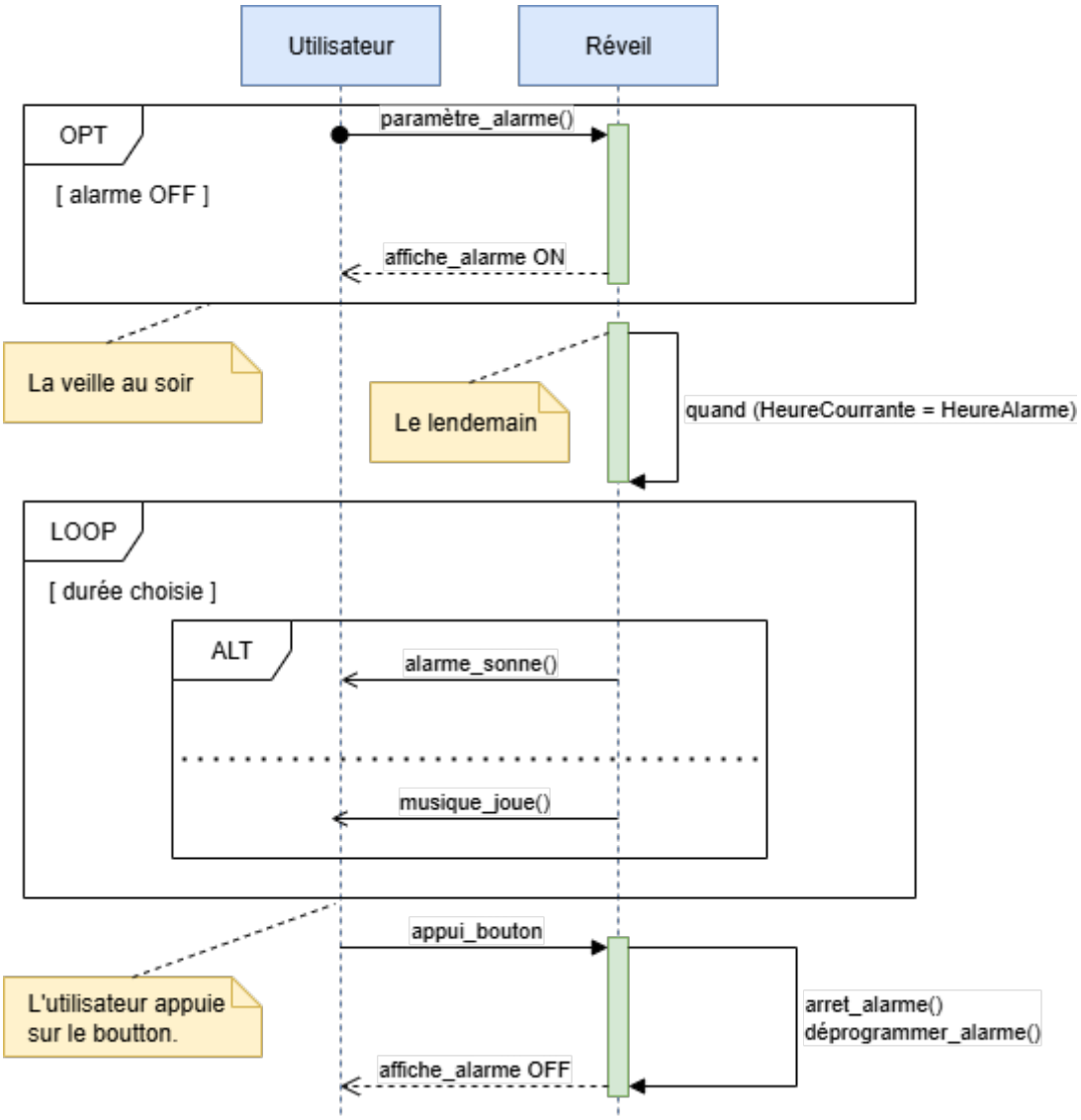


Enfin, il est possible de modéliser certaines actions conditionnelles ou des boucles en encapsulant l'échange dans un bloc.

Ce bloc pourra être nommé :

- OPT : ***Optionnel***. Ce fragment ne s'exécute que si une condition est vraie
- LOOP : ***Boucle***. Ce fragment peut s'exécuter plusieurs fois ou jusqu'a ce qu'une condition soit vraie
- ALT : ***Alternatif***. Ce fragment s'exécute si une condition est vraie suivie de sinon : la condition est fausse

Ainsi, la séquence de réveil se construira comme suit :



Revision #2
Created 2026-07-17 22:14:53 UTC by Olivier
Updated 2026-08-12 13:26:51 UTC by Olivier